UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/727,327	11/29/2000	Marian Montequinfela MacCormack	50886-03USPX	2273

7590 12/30/2003

Andre M. Szuwalski
Jenkens & Gilchrist, P.C.
Suite 3200
1445 Ross Avenue
Dallas, TX 75202-2799

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	7

DATE MAILED: 12/30/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/727,327	MACCORMACK, MARIAN MONTEQUINFELA	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 November 2000*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-19* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-19* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *29 November 2000* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) *4,5,6*.
4) ☐ Interview Summary (PTO-413) Paper No(s). _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

1. This action is responsive to the application filed November 29, 2000.

Claims 1-19 have been submitted for examination.

Claim Objections

2. Claim 11 is objected to for a minor informality: element "a relocation instructions" (line 3) should be – a relocation instruction – without an "s".

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-4, 6-17, and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Cahill et al., USPN: 6,021,272 (hereinafter Cahill), in view of Brooks et al., USPN: 5,819,097 (hereinafter Brooks).

As per claim 1, Cahill discloses a method of generating source code listing from an object code sequence comprising a section data including a plurality of program instructions, said section associated a relocation section including at least one relocation instruction (e.g. *tag table 218, text relocs 305 , data relocs 306* - Fig. 3) used at link time to modify the object code to generate an executable program (Fig. 1 – Note: linker and object code implicitly discloses modify object code at link time to generate executable), the method comprising:

for each location in the section data determining if that location has a relocation instruction associated therewith (e.g. *relocation tags* - col. 3, lines 18-25; col. 7, lines 9-35; *registers must be reallocated, checkpoint* - col. 12, line 25 to col. 13, line 27; Fig. 8-9);

deriving relocation additional information concerning the section data (e.g. *tags* - col. 7, lines 25-35); and

generating the reversed object code listing for that object code modification instruction and additional information, such listing comprising the modified object code that was derived from said additional information (e.g. Fig. 2, 7A-D).

But Cahill does not explicitly specify reading said relocation instruction identified for each location in the section data. However, Cahill discloses disassembling process to recompose the original object code based on verifying and readjust each instruction-related location data, and *jump table* relocation instruction, such modification derived from additional information like tag, target address, target label (e.g. Fig. 4-5; col. 9, line 45 to col. 10, line 53). Hence Cahill has implicitly disclosed reading relocation instruction identified at a location in the section data.

Nor does Cahill specify generating the source code listing in lieu of reversed object code for the location in the section data, such listing being derived from said relocation instruction and additional relocation information. In a method analogous to that of Cahill using decompilation or unlinking of a compiled object program (*object program 24, unlinked object 54* -Fig. 2), Brooks discloses unlinking the object program back into source code (e.g. Fig. 5) using additional information such as memory address reallocated values (or relocation instructions), instruction and symbol table, and tags values (e.g. col. 7, line 37 to col. 8, line 8; col. 8, line 66 to col. 9, line 21; Fig. 6) similar to the relocation additional information as taught by Cahill. It

would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the disassembling process as taught by Cahill so that the recreation of source code is based on relocation identification and additional relocation information as disclosed by Brooks, because such source code being recreated can serve as human-remodifiable and reusable program and obviating compiler resources because the source code thus re-generated would not require making change to the compiler or the hardware controller (Brooks: col. 1, lines 39-59; col. 2, lines 50-59).

As per claim 2, this source code limitation would have been obvious in view of the rationale used in claim 1 above using Brooks' teaching.

As per claims 3-4, Cahill discloses object code sequence being part of the object code module (e.g. col. 3, lines 18-25) and part of an executable program (Fig. 1, 6; col. 12, line 25 to col. 13, line 27 – Note: only checkpoint labels inserted for reallocation and error-checking purpose implicitly discloses relocation instructions being a part of the executable)

As per claim 6, Cahill discloses relocation instructions for performing calculations using a stack (e.g. Fig. 6; col. 12, Table 6-1).

As per claim 7, Cahill discloses arithmetic expression from branch instruction generated from source code (e.g. col. 13, lines 18 to col. 14, line 34 – Note: since object code is inherently derived from original source code, branch instructions arithmetic being defined from source code are implicitly disclosed).

As per claim 8, Cahill discloses assembler directives (e.g. *PIAI, checkpoint instructions* – col. 11-12; col. 12, table 6-1, 6-2; assembly code instruction *400* – Fig. 4).

As per claim 9, Cahill discloses operands values defined in the source code (e.g. col. 9, lines 12-36 Note: since object code is inherently derived from original source code, operands instructions *tuples* being defined from source code are implicitly disclosed).

As per claim 10, Cahill disclose event information (e.g. *signal handler* – col. 15, line 35 to col. 16, line 4).

As per claim 11, Cahill discloses a lister for generating reversed object code listing from an object code sequence comprising program instructions associating a relocation instruction therewith; the lister comprising:

a instruction reader (e.g. col. 6, lines 55-65);

a relocation reader (*relocation tags* - col. 3, lines 18-25; col. 7, lines 9-35; *registers must be reallocated, checkpoint* - col. 12, line 25 to col. 13, line 27; Fig. 8-9);

a relocation identifier (*relocation tags* - col. 3, lines 18-25);

a disassembler for disassembling said program instructions to generate reversed object code and disassembling additional information (e.g. *tags* - col. 7, lines 25-35) from said relocation instruction (Fig. 2, 7A-D).

But Cahill does not specify generating the source code display in lieu of reversed object code from disassembling program instructions, such source code listing being derived from said relocation instruction and additional relocation information and being in human-readable form. But this limitation has been addressed in claim 1 above.

As per claim 12, Cahill discloses offset determining means for determining offset from relocation instruction (e.g. col. 9, lines 12-16; col. 12, table 6-2 – Note: offset information implicitly discloses existence of program count detecting means).

As per claim 13, Cahill discloses a type field (e.g. *Type 430* - Fig. 4).

As per claim 14, Cahill discloses an operand, an operator (e.g. *tuple* – col. 9, lines 9-21; *jump table* – col. 9, line 40 to col. 10, line 54 – Note: branch instructions inherently includes operands and operator for address adjustment),

an event (col. 15, line 35 to col. 16, line 4); and

a directive (e.g. *PIAI, checkpoint instructions* – col. 11-12; col. 12, table 6-1, 6-2; assembly code instruction *400* – Fig. 4).

As per claim 15, see claim 7 rejection.

As per claim 16, see rejection of claim 6.

As per claim 17, Cahill discloses selecting of event handler in resolving reallocation of addresses (*signal handler* – col. 15, line 35 to col. 16, line 4) but does not expressly disclose event calculator. However, Cahill discloses selectively calling of handler according to set bit (e.g. Fig. 7c); hence has implicitly disclosed a form of evaluating which event handler to invoke.

As per claim 19, see claim 8 for corresponding rejection.

5. Claim 1-4, 6-17, and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Brandes, USPN: 5,946,484 (hereinafter Brandes), in view of Noda et al., JPPN: JP408147155 A (hereinafter Noda).

Cahill discloses a method of generating source code listing from an object code sequence comprising a section data including a plurality of program instructions, said section associated a relocation section including at least one relocation instruction (e.g. *tag table 218, text relocs 305 , data relocs 306* - Fig. 3) used at link time to modify the object code to generate an executable

program (Fig. 1 – Note: linker and object code implicitly discloses modify object code at link time to generate executable), the method comprising:

determining (location has a relocation instruction);

deriving (relocation additional information); and

generating (reversed object code), such limitations having been addressed in claim 1 above.

However, Cahill does not explicitly specify reading said relocation instruction identified in the section data. But this limitation has been addressed in claim 1 above.

Nor does Cahill specify generating the source code listing in lieu of reversed object code for the location in the section data, such listing being derived from said relocation instruction and additional relocation information. Using labels for additional information similar to Cahill's tag table, Noda, in a method to disassemble assembler code, discloses utilizing relocation instruction to inversely assemble source code based on the modification performed when using those labels addresses (Abstract; Constitution).

As per claim 2, this limitation would have been obvious in view of the rationale used in claim 1 above using Noda's source code teachings.

As per claims 3-4, see rejection in claims 3-4 from USC 103(a) section from above using Cahill and Brooks.

As per claim 6, see claim 6 rejection from USC 103(a) section from above using Cahill and Brooks.

As per claims 7-10, see respective claims 7-10 rejections from USC 103(a) section from above using Cahill and Brooks.

As per claim 11, Cahill discloses a lister for generating reversed object code listing from an object code sequence comprising program instructions associating a relocation instruction therewith; the lister comprising:

a instruction reader (e.g. col. 6, lines 55-65);

a relocation reader (*relocation tags* - col. 3, lines 18-25; col. 7, lines 9-35; *registers must be reallocated, checkpoint* - col. 12, line 25 to col. 13, line 27; Fig. 8-9);

a relocation identifier (*relocation tags* - col. 3, lines 18-25);

a disassembler for disassembling said program instructions to generate reversed object code and disassembling additional information (e.g. *tags* - col. 7, lines 25-35) from said relocation instruction (Fig. 2, 7A-D).

But Cahill does not specify generating the source code display in lieu of reversed object code listing from disassembling program instructions, such source code listing being derived from said relocation instruction and additional relocation information and being in human-readable form. But this limitation has been addressed in claim 1 from USC 103(a) section from above using Cahill and Noda.

As per claims 12-17, and 19, see respective claims 12-17, and 19 rejections from USC 103(a) section from above using Cahill and Brooks.

6. Claim 5 is rejected under 35 U.S.C. 103(a) as being unpatentable over Cahill et al., USPN: 6,021,272 and Brooks et al., USPN: 5,819,097; as applied to claim 1; and further in view of Brandes, USPN: 5,946,484 (hereinafter Brandes).

As per claim 5, Cahill does not specify a library of object code holding a number of frequently used code sequences together with their associated relocations. But Cahill discloses

means for preventing contention in re-modifying already modified libraries (library 170 – Fig. 1; col. 15, lines 12-45) and Brooks discloses an instruction table to expedite reallocation of instructions data at runtime (Fig. 3,5); thus suggesting a means to alleviate extraneous usage of resources by avoiding re-processing of entries or instructions already addressed or scanned during the linking or assembling process, i.e. marking objects, entries/steps so that extraneous re-processing of those objects or entries/steps is a well-known concept in computer processing, when such processing involves iterative or highly repetitive executing steps. Likewise, in a method to regenerate source program from an assembled object code format analogous to the method by Cahill/Brooks, Brandes discloses processing a plurality of object code format instructions and comparing them against pre-recorded patterns (e.g. col. 13, lines 37-64) and elimination of superfluous generation of source file (*NOCMT*- col. 27, lines 23-55). The recording of frequently encountered patterns of object code instructions and avoiding of recreating of superfluous code is suggested. In view of such teachings by Brandes, it would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the object module storage associated with relocation information by Cahill/Brooks so to additionally store object code in library for reuse using the concept of storing most frequently used patterns of code as suggested by Brandes. One of ordinary skill in the art would be motivated to do so because that way frequently encountered object code format and instruction structures are stored in reusable library form would alleviate extraneous use of resources as observed above.

7. Claim 5 is rejected under 35 U.S.C. 103(a) as being unpatentable over Cahill et al., USPN: 6,021,272 and Noda et al., JPPN: JP408147155 A; as applied to claim 1; and further in view of Brandes, USPN: 5,946,484.

As per claim 5, Noda discloses using tags to expedite processing of object code during disassembling process (Abstract; Constitution) and Cahill discloses alleviating resources contention (col. 15, lines 12-45), both hence suggesting marking objects, entries/steps so that extraneous re-processing of those objects or entries/steps, a well-known concept in computer processing, when such processing involves iterative or highly repetitive executing steps. Further, Brandes discloses processing a plurality of object code format instructions and comparing them against pre-recorded patterns (e.g. col. 13, lines 37-64) and elimination of superfluous generation of source file (*NOCMT*- col. 27, lines 23-55). The limitation to include most frequently used code in library object code is rejected herein using the rationale for combining with Brandes for the same motivation as set forth in claim 5 from USC 103(a) section from above using Cahill and Brooks.

8. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Cahill et al., USPN: 6,021,272 and Brooks et al., USPN: 5,819,097, as applied to claim 17; and further in view of Clarke et al., USPN: 5,754,759 (hereinafter Clarke).

As per claim 18, Cahill discloses tag stack and event handler selection as addressed in claims 6 and 17; but does not explicitly discloses event stack. Clarke, in a method to monitor program debug execution using disassembly module (col. 6, line 12-25) similar to Cahill's error-check of object modules at run-time, discloses a event stack (col. 9, lines 49-67). In view of the teachings by Cahill from above, it would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the use of stack and signal handler by Cahill so to implement an event stack as taught by Clarke because use of stack as taught by Clarke for storage and immediate resolution of instantaneous occurring of signal handler requests as

suggested by Cahill would enable fulfilling of a larger amount of distress calls incurred in a system where error-checking involving memory address readjustment is as crucial as suggested in the relocation execution by Cahill.

9. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Cahill et al., USPN: 6,021,272 and Noda et al., JPPN: JP408147155 A, as applied to claim 17; and further in view of Clarke et al., USPN: 5,754,759.

As per claim 18, see rationale of claim 18 from 35 U.S.C. 103(a) with Cahill et al. and Brooks et al., USPN: 5,819,097, in view of Clarke from above.

Conclusion

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Pat No. 6,151,701 to Humphreys et al., disclosing stack pushing and decompiling using hook codes.

U.S. Pat No. 5,005,152 to Knutsen, disclosing jump tables and matrix and index arithmetics.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please label "PROPOSED" or "DRAFT" – Please consult Examiner before use)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

VAT
December 28, 2003

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100